Claims

What is claimed is:

1. A system for transforming XML items, the system comprising:

   a transformer adapted to transform one or more input XML items in a first format to one or more transformed XML items in one or more second formats; and

   an output manager adapted to facilitate selectively pulling and/or pushing a subset of the transformed XML items.

2. The system of claim 1, where the transformer comprises an action frame stack adapted to hold one or more actions, an event state machine adapted to track state associated with transforming the one or more XML items and an event processor adapted to receive events generated in processing the one or more actions stored in the action frame stack.

3. The system of claim 1, further comprising a compiler adapted to compile one or more style sheets and produce one or more actions that can be employed by the transformer in processing associated with transforming the one or more input XML items.

4. The system of claim 3, where the compiler is further adapted to resolve one or more external references in the one or more style sheets.

5. The system of claim 4, where the input XML items are input from one or more data stores.

6. The system of claim 5, further comprising an input abstracter adapted to expose data stored in the one or more data stores in a common representation.

7. The system of claim 6, where the input abstractor is further adapted to abstract a reference to a node within an XPath document.

8. The system of claim 7 where the input abstractor is further adapted to expose the data stored in the one or more data stores as a data model and infoset.

9. The system of claim 8, where the input abstractor is further adapted to provide a cursor model over data stored in a data store to facilitate presenting a stream of nodes to the transformer.

10. The system of claim 9, where the input abstractor is further adapted to provide a virtual node that can be employed to traverse the stream of nodes.

11. The system of claim 10 where the input abstractor is an XPathNavigator.

12. The system of claim 6, further comprising a node selection abstractor adapted to dynamically construct a subset of input XML items from a set of input XML items, where the subset of input XML items are responsive to a query.

13. The system of claim 12 where the node selection abstractor is further adapted to facilitate navigating the subset of input XML items.

14. The system of claim 13 where the node selection abstractor is an XPathNodeIterator.

15. The system of claim 12, further comprising an optimized data store adapted to store one or more XML items in a manner that facilitates minimizing processing associated with constructing the subset of input XML items *via* a query.

16. The system of claim 15 where the optimized data store stores data in a data representation format that facilitates optimizing an XPath query.

17. The system of claim 16, where the data representation format comprises expanded XML entities, deleted XML declarations and DOM model data converted to XPath model data.

18. The system of claim 17, where the optimized data store is an XPathDocument.

19. A computer readable medium storing computer executable components of a system for transforming XML items, the system comprising:

a transforming component adapted to transform an input XML item from a first format to a transformed XML item in one or more second formats;

an output managing component adapted to facilitate selectively pulling and/or pushing a subset of the transformed XML items;

a compiling component adapted to compile a style sheet and to produce one or more actions that can be employed by the transforming component in processing associated with transforming the input XML item;

an input abstracting component adapted to present input XML items stored in one or more different representations to the transforming component in a common representation; and

a node selection abstracting component adapted to dynamically construct a subset of input XML items from a set of input XML items, where the subset of input XML items are responsive to a query.

20. A method for transforming XML items, the method comprising:

inputting one or more style sheets;

compiling the one or more style sheets to produce one or more actions;

selectively inputting one or more XML items;

pattern matching the one or more XML items to one or more templates located in the one or more style sheets;

selectively performing transformations on a subset of the one or more XML items based, at least in part, on one or more actions associated with the one or more templates; and building an output record of one or more transformed XML items, where the output record may be pushed to a destination data source and/or pulled by a destination data source.

21. The method of claim 20, where compiling the one or more style sheets comprises:
resolving one or more external references in the one or more style sheets;
identifying a root action;
compiling the root action;
identifying one or more non-root actions that descend from the root action; and
compiling the one or more non-root actions.

22. The method of claim 21, where compiling the root action comprises:
compiling root action attributes; and
verifying attributes.

23. The method of claim 22, where compiling the one ore more non-root actions comprises:
compiling attributes;
verifying attributes; and
recursively compiling a body to the non-root action.

24. The method of claim 23, where compiling attributes comprises:
storing one or more attribute names in memory;
storing one or more attribute values in memory; and
adding one or more queries to a query store.

25. The method of claim 20, where the one or more XML items are input from one or more input data stores.

26. The method of claim 25, where the one or more input data stores store the one or more XML items in one or more different data representations.

27. The method of claim 26, where the one or more different data representations are normalized to a common data representation that exposes a data model and infoset.

28. The method of claim 27, where the one or more input XML items are read from a stream of input XML items.

29. The method of claim 28, where the one or more input XML items are selectively read from a stream of XML items.

30. The method of claim 20 where selectively performing transformations on a subset of the one or more XML items comprises:

initializing a transformer;
pushing a root action onto an action frame stack;
performing a template lookup action for the root of a style sheet;
pushing one or more actions onto the action frame stack;
receiving one or more instructions to execute an action;
performing one or more actions;
generating one or more events;
validating the one or more events;
sending one or more events and associated content to a record builder; and
popping the one or more actions off the action frame stack.

31. The method of claim 30, where performing the one or more actions comprises:

selectively adding one or more markup characters to a data item;
selectively deleting one or more markup characters from a data item;
selectively adding one or more content characters to a data item;
selectively deleting one or more content characters from a data item; and

selectively generating an output item from the data item.

32. The method of claim 20 where building an output record of transformed XML items comprises:

receiving an event;

validating a content associated with the event; and

selectively adding the content to an output record.

33. A computer readable medium storing computer executable instructions operable to perform the method of claim 20.

34. A system for transforming XML items, the system comprising:

means for receiving one or more files containing XML item pattern matching rules and associated output generating rules;

means for compiling the one or more files to produce one or more compiled pattern matching rules and output generating rules;

means for inputting one or more XML items;

means for applying the compiled pattern matching rules to the one or more XML items to identify one or more XML items to alter;

means to alter the one or more identified XML items; and

means for building an output record of one or more transformed XML items by applying the compiled output generating rules.

35. A data packet adapted to be transmitted between two or more computer processes, the data packet comprising:

one or more first fields adapted to store an input XML item in an abstracted format; and

one or more second fields adapted to store metadata associated with the abstracted input XML item.

36. The data packet of claim 35, where the abstracted format conforms to the XPath specification.

37. The data packet of claim 35, where the metadata exposes the W3C Infoset concerning the input XML item.